MIME Protocol Module for TTCN-3 Toolset with TITAN, Description

Eduárd Czimbalmos

Version 1551-CNL 113 352, Rev. A, 218-06-052

Table of Contents

Abstract

The purpose of this document is to specify the content of The MIME protocol module. [1]

About This Document

How to Read This Document

This is the User Guide for the MIME protocol module. The MIME protocol module is developed for the TTCN-3 Toolset with TITAN.

Presumed Knowledge

To use this protocol module the knowledge of the TTCN-3 language [2] is essential.

Functionality

Protocol VersionIimplemented

This set of protocol modules implements protocol messages and constants of the MIME protocol, (see [1]) with the modifications specified here.

Ericsson-specific Changes

There is no Ericsson specific change in this product.

Backward Incompatibilities

None.

Protocol Modifications/Deviations

Protocol modules contain the following modifications/deviations from [1] changing the protocol message structure/behaviour:

None.

System Requirements

Protocol modules are a set of TTCN-3 source code files that can be used as part of TTCN-3 test suites only. Hence, protocol modules alone do not put specific requirements on the system used. However in order to compile and execute a TTCN-3 test suite using the set of protocol modules the following system requirements must be satisfied:

- TITAN TTCN-3 Test Executor R7A (1.7.pl0) or higher installed. For installation guide see [4]. Please note: This version of the protocol module is not compatible with TITAN releases earlier than R7A.

Feature list

Encoding/Decoding and Other Related Functions

This product contains encoding/decoding functions, which assure correct encoding of messages when sent from Titan and correct decoding of messages when received by Titan. Implemented encoding/decoding functions:

Functions to be used for decoding MIME bodies with text contents

```
external function f_MIME_Encode(in PDU_MIME_entity inent) return charstring;
external function f_MIME_Decode(in charstring inent) return PDU_MIME_entity;
external function  f_MIME_build_multipart(in PDU_MIME_entity_list entities, in
charstring delimiter) return charstring;
external function  f_MIME_get_multipart(in PDU_MIME_entity inent)
return PDU_MIME_entity_list;
```

Functions to be used for decoding MIME bodies with binary contents

```
external function f_MIME_Encode_binary(in PDU_MIME_entity_binary inent) return
octetstring;
external function f_MIME_Decode_binary(in octetstring inent) return
PDU_MIME_entity_binary;
external function f_MIME_build_multipart_binary(in PDU_MIME_entity_list_binary
entities, in charstring delimiter) return octetstring;
external function  f_MIME_get_multipart_binary(in PDU_MIME_entity_binary inent)
return PDU_MIME_entity_list_binary;
```

Other functions

```
external function  f_MIME_Base64_Encode(in octetstring inent) return charstring;
external function  f_MIME_Base64_Decode(in charstring inent) return octetstring;
```

Example Usage

```
module MIME_Example {

import from MIME_Types all;

const charstring  c_example_MIME_Multipart_ContentType := "Content-Type:
multipart/mixed; boundary=\"----=_Part_0_1167317508.1518033268011\"\r\n\r\n";

type component Comp {}

testcase tc_encodeDecodeMIME_binary() runs on Comp {

  // Encoding:
  var PDU_MIME_entity_list_binary vl_list;
  vl_list[0] := {
    content_type := { content_type := "message", subtype := "cpim", parameters := omit
},
    content_encoding := omit,
    other_fields := omit,
    payload := char2oct("To: <tel:+112009000722>\r\nFrom:
<tel:+112009000721>\r\nDateTime: 18020714542700\r\nNS: imdn
<urn:ietf:params:imdn>\r\n\r\nimdn.Message-ID: 2359167681518033267971\r\n\r\nContent-
Type: text/plain; charset=utf-8\r\nContent-Length: 27\r\n\r\n*****NOT
INTERWORKABLE*****")
  };
  vl_list[1] := {
    content_type := { content_type := "application", subtype := "vnd.3gpp.sms",
parameters := omit },
    content_encoding := omit,
    other_fields := omit,
    payload := 'AABBCCDD'O
  };
  var octetstring vl_encoded := f_MIME_build_multipart_binary(vl_list, "----
=_Part_0_1167317508.1518033268011");

  // Decoding:
  var PDU_MIME_entity_binary vl_mime :=
f_MIME_Decode_binary(char2oct(c_example_MIME_Multipart_ContentType));
  vl_mime.payload := vl_encoded;
  vl_list := f_MIME_get_multipart_binary(vl_mime);
  if(vl_list[1].payload == 'AABBCCDD'O) {
    setverdict(pass, "The binary payload matches the expected!");
  }

}

}
```

Protocol Modules

Overview

Protocol modules implement the message structures of the related protocol in a formalized way, using the standard specification language TTCN-3 . This allows defining of test data (templates) in the TTCN-3 language [6] and correctly encoding/decoding messages when executing test suites using the Titan TTCN-3 test environment [4].

Installation

The set of protocol modules can be used in developing TTCN-3 test suites using any text editor. However to make the work more efficient a TTCN-3-enabled text editor is recommended (e.g. Eclipse, nedit, xemacs). Since the MIME protocol is used as a part of a TTCN-3 test suite, this requires TTCN-3 Test Executor be installed before the module can be compiled and executed together with other parts of the test suite. For more details on the installation of TTCN-3 Test Executor see the relevant section of [3].

Release of the protocol module contains the *MIME_CNL113352.tpd* file, which can be used to generate a Makefile with `ttcn3_makefilegen`, or import the project into Eclipse with the TITAN Designer plugin.

Configuration

None.

Parser generation rules

In order to generate the *.c* and *.h* files from *.y* and *.l* the following Makefile rules should be used:

```
MIME_parse_.tab.c MIME_parse_.tab.h: MIME_parse.y
    bison -dv -p MIME_parse_ -b MIME_parse_ $<

lex.MIME_parse_.c: MIME_parse.l
    flex -Cfr -8 -Bvpp -PMIME_parse_ MIME_parse.l
```

The *.h* and *.c* parser files should be generated during the protocol module development. Only the pregenerated files are needed for test case development and test execution.

Terminology

No specific terminology used.

Abbreviations

MIME	Multipurpose Internet Mail Extensions
TTCN-3	Testing and Test Control Notation version 3

References

[[1]] RFC 2045 and RFC 2046
Multipurpose Internet Mail Extensions (MIME) Part One and Part Two

[2] ETSI ES 201 873-1 v4.8.1 (2016-07) Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: Core Language

[3] 1/ 198 17-CRL 113 200/6 Uen User Guide for TITAN TTCN-3 Test Executor